L 12

```
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS   YYY         YYY     SSSSSSSSSSSS
SSS               YYY         YYY  SSS
SSS               YYY         YYY  SSS
SSS               YYY         YYY  SSS
SSS                  YYY   YYY     SSS
SSS                  YYY   YYY     SSS
SSS                  YYY   YYY     SSS
   SSSSSSSSS            YYY           SSSSSSSSS
   SSSSSSSSS            YYY           SSSSSSSSS
   SSSSSSSSS            YYY           SSSSSSSSS
            SSS         YYY                    SSS
            SSS         YYY                    SSS
            SSS         YYY                    SSS
            SSS         YYY                    SSS
            SSS         YYY                    SSS
            SSS         YYY                    SSS
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS
SSSSSSSSSSSS            YYY        SSSSSSSSSSSS
```

```
**FILE**ID**SYSBRKTHR                          I 16



  SSSSSSSS  YY      YY  SSSSSSSS  BBBBBBBB    RRRRRRRR    KK      KK  TTTTTTTTTT  HH      HH  RRRRRRRR
  SSSSSSSS  YY      YY  SSSSSSSS  BBBBBBBB    RRRRRRRR    KK      KK  TTTTTTTTTT  HH      HH  RRRRRRRR
SS          YY      YY SS         BB      BB  RR      RR  KK      KK      TT      HH      HH  RR      RR
SS          YY      YY SS         BB      BB  RR      RR  KK      KK      TT      HH      HH  RR      RR
SS            YY  YY   SS         BB      BB  RR      RR  KK    KK        TT      HH      HH  RR      RR
SS            YY  YY   SS         BB      BB  RR      RR  KK    KK        TT      HH      HH  RR      RR
  SSSSSS        YY       SSSSSS   BBBBBBBB    RRRRRRRR    KKKKKK          TT      HHHHHHHHHH  RRRRRRRR
  SSSSSS        YY       SSSSSS   BBBBBBBB    RRRRRRRR    KKKKKK          TT      HHHHHHHHHH  RRRRRRRR
        SS      YY             SS BB      BB  RR  RR      KK    KK        TT      HH      HH  RR  RR
        SS      YY             SS BB      BB  RR  RR      KK    KK        TT      HH      HH  RR  RR
        SS      YY             SS BB      BB  RR    RR    KK      KK      TT      HH      HH  RR    RR      ....
        SS      YY             SS BB      BB  RR    RR    KK      KK      TT      HH      HH  RR    RR      ....
SSSSSSSS        YY     SSSSSSSS   BBBBBBBB    RR      RR  KK      KK      TT      HH      HH  RR      RR  ....
SSSSSSSS        YY     SSSSSSSS   BBBBBBBB    RR      RR  KK      KK      TT      HH      HH  RR      RR  ....


LL          IIIIII     SSSSSSSS
LL          IIIIII     SSSSSSSS
LL            II     SS
LL            II     SS
LL            II     SS
LL            II     SS
LL            II       SSSSSS
LL            II       SSSSSS
LL            II             SS
LL            II             SS
LL            II             SS
LL            II             SS
LLLLLLLLLL  IIIIII   SSSSSSSS
LLLLLLLLLL  IIIIII   SSSSSSSS
```

```
0000          .TITLE  SYSBRKTHR - Write breakthru to terminals
0000          .IDENT  'V04-000'
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ;
0000    ; FACILITY:
0000    ;
0000    ;       SYS
0000    ;
0000    ; INCLUDES:
0000    ;       $BRKTHRU system service
0000    ;       $BRDCST system service
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       Write breakthru message to specified terminals and mailboxes.
0000    ;
0000    ; ENVIRONMENT:
0000    ;
0000    ;       Kernel Mode. IPL 0 and 2.
0000    ;
0000    ;--
0000    ;
0000    ; AUTHOR: Jake VanNoy, CREATION DATE: 3-Feb-1983
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-011 JLV0392         Jake VanNoy                     26-JUL-1984
0000    ;               Make check for TRM and SPL at HAVE_UCB.
0000    ;               Do not write message to mailbox if class disabled.
0000    ;
0000    ;       V03-010 JLV0347         Jake VanNoy                     8-APR-1984
0000    ;               Skip terminal if NET is set.  Fix problem in
```

```
0000     ;                    check for broadcast to same username.
0000     ;                    Copy DEVNAME to SENDNAME so that cluster broadcast
0000     ;                    to device will work.  Change MOVC of device name
0000     ;                    fields to MOVQ's.
0000     ;
0000     ;           V03-009 JLV0339         Jake VanNoy              9-MAR-1984
0000     ;                    Skip terminal if PASSALL is set.  Fix mailbox message
0000     ;                    to have just DDC part of device name.  Force timeout
0000     ;                    of a cluster breakthru request to 15 seconds on all
0000     ;                    nodes except local.  Fix bug that used BRK$L_FLAGS as
0000     ;                    scratch.
0000     ;
0000     ;           V03-008 ACG0385         Andrew C. Goldstein,     28-Dec-1983  15:27
0000     ;                    Change UAF$S_USERNAME use to JIB$S_USERNAME, due to
0000     ;                    pending UAF format changes
0000     ;
0000     ;           V03-007 JLV0308         Jake VanNoy              22-SEP-1983
0000     ;                    Complete work started in JLV0307. Fix check against
0000     ;                    username in GET_SENDTO.  Change parameter in call
0000     ;                    to IOC$CVT_DEVNAM, since the interface to that routine
0000     ;                    has changed.
0000     ;
0000     ;           V03-006 JLV0307         Jake VanNoy              7-SEP-1983
0000     ;                    Fix enhanced privilege bug. Wait until after cluster
0000     ;                    broadcast to deallocate BRK. Fix bug in defaulting of
0000     ;                    carriage control in $BRDCST. Add use of EXE$SIGTORET
0000     ;                    in $BRDCST.
0000     ;
0000     ;           V03-005 JLV0302         Jake VanNoy              22-AUG-1983
0000     ;                    Add MOVC5 to zero entire BRK structure up to where text
0000     ;                    is placed. This allowed removing separate CLRx instructions
0000     ;                    in initialization. Save register around MOVC in GET_SENDTO.
0000     ;                    Change exit path for SS$_NOOPER error code.
0000     ;
0000     ;           V03-004 JLV0300         Jake VanNoy              30-JUL-1983
0000     ;                    Add OPER priv checks. Allow $BRKTHRU to same username
0000     ;                    without priv.  Initialize mailbox prefix code. Remove
0000     ;                    BRK$ symbols from here and move them to LIB. This
0000     ;                    allows cluster broadcast code to use BRK structure.
0000     ;                    Add IO$M_CANCTRLO to QIO.  Make use of IOC$CVT_DEVNAM.
0000     ;
0000     ;           V03-003 LJK0213         Lawrence J. Kenah        23-Jun-1983
0000     ;                    Unlock data base before calling GET_NEXT_TERMINAL to make
0000     ;                    sure that $GETJPI is not called at IPL 2.
0000     ;
0000     ;           V03-002 JLV0269         Jake VanNoy              27-MAY-1983
0000     ;                    Fix bugs in SET_PRIV routine. Add code to use REQID.
0000     ;                    Add code to call EXE$CSP_BRKTHRU, the cluster broadcast
0000     ;                    routine.
0000     ;
0000     ;           V03-001 JLV0245         Jake VanNoy              29-APR-1983
0000     ;                    First pass cleanup. Include code for EXE$BRDCST here,
0000     ;                    this obsoletes the old SYSBRDCST module.
0000     ;
0000     ;**
0000
0000
```

```
                     0000            .SBTTL  DECLARATIONS
                     0000      ;
                     0000      ; INCLUDE FILES:
                     0000      ;
                     0000            $BRKDEF                          ; Define BRKTHRU interface symbols
                     0000            $BRKTDEF                         ; Define BRK block
                     0000            $CCBDEF                          ; Define channel control block
                     0000            $DDBDEF                          ; Define device data block
                     0000            $DEVDEF                          ; Define device symbols
                     0000            $DVIDEF                          ; Define GETDVI symbols
                     0000            $IODEF                           ; Define I/O request symbols
                     0000            $IPLDEF                          ; Define IPL fields
                     0000            $JIBDEF                          ; Define Job Information Block
                     0000            $JPIDEF                          ; Define GETJPI symbols
                     0000            $PCBDEF                          ; Define process control block
                     0000            $PHDDEF                          ; Define process header
                     0000            $PRVDEF                          ; Define privilege names
                     0000            $PSLDEF                          ; Define PSL fields
                     0000            $SSDEF                           ; Define status codes
                     0000            $TTDEF                           ; Define tt devdepend symbols
                     0000            $TT2DEF                          ; Define tt devdepnd2 symbols
                     0000            $TTYUCBDEF                       ; terminal ucb extensions
                     0000            $UAFDEF                          ; Define user authorization symbols
                     0000            $UCBDEF                          ; Define UCB
                     0000      ;
                     0000      ; MACROS:
                     0000      ;
                     0000
                     0000      ;
                     0000      ; EQUATED SYMBOLS:
                     0000      ;
                     0000
          00000004   0000      EFN      = 4
          00000008   0000      MSGBUF   = 8
          0000000C   0000      SENDTO   = 12
          00000010   0000      SENDTYPE = 16
          00000014   0000      IOSB     = 20
          00000018   0000      CARCON   = 24
          0000001C   0000      FLAGS    = 28
          00000020   0000      REQID    = 32              ;
          00000024   0000      TIMOUT   = 36
          00000028   0000      ASTADR   = 40
          0000002C   0000      ASTPRM   = 44
                     0000
          0000001F   0000      BRK_C_JPIEFN    = 31     ; system efn
          0000001F   0000      BRK_C_TIMEFN    = 31
          0000001F   0000      BRK_C_QIOEFN    = 31
          0000001F   0000      BRK_C_DVIEFN    = 31
          0000001F   0000      BRK_C_BRDCSTEFN = 31
          00000004   0000      BRK_C_MINTIME   = 4      ; minimum time in seconds
          00000004   0000      BRK_C_SIMULCAST = 4      ; simultaneous QIO's
          00000018   0000      BRK_C_MAXLINES  = 24     ; maximum number of lines allowed to clear in screen write
          0000000F   0000      BRK_C_CLUTIMEOUT= 15     ; forced timeout for cluster broadcast
                     0000
          20000000   0000      PRV$M_BYPASS    = 1@PRV$V_BYPASS         ; define mask
          80000000   0000      PRV$M_SHARE     = 1@PRV$V_SHARE          ; define mask
                     0000
```

```
0000      ; following assumes for MOVQ's of name buffer's
0000
0000      ASSUME DDB$S_NAME        EQ 16
0000      ASSUME BRK$S_DEVNAME     EQ 16
0000      ASSUME BRK$S_SENDNAME    EQ 16
0000      ASSUME BRK$S_TRMNAME     EQ 16
0000
```

```
                                       0000     ;
                                       0000     ; Local storage offsets for temporary stack allocation
                                       0000     ;
                                       0000
                                       0000     ;
                                       0000     ; getjpi stack items
                                       0000     ;
                                       0000             $DEFINI STK
                                       0000
                                       0000     $DEF    STK$W_USERSIZ   .BLKW
                                       0002     $DEF    STK$W_USERJPI   .BLKW
                                       0004     $DEF    STK$L_USERNAME  .BLKL
                                       0008     $DEF    STK$L_USERLENR  .BLKL
                                       000C
                                       000C     $DEF    STK$W_TERMSIZ   .BLKW
                                       000E     $DEF    STK$W_TERMJPI   .BLKW
                                       0010     $DEF    STK$L_TERMNAME  .BLKL
                                       0014     $DEF    STK$L_TERMLENR  .BLKL
                                       0018
                                       0018     $DEF    STK$L_ENDLIST   .BLKL
                                       001C
                                       001C     $DEF    STK$W_USERLEN   .BLKW
                                       001E     $DEF    STK$T_USERNAME  .BLKB   JIB$S_USERNAME
                                       002A     $DEF    STK$W_TERMLEN   .BLKW
                                       002C
                                       002C     $DEF    STK$C_LEN
                                       002C
                                       002C             $DEFEND STK
                                       0000     ;
                                       0000     ; OWN STORAGE:
                                       0000     ;
                                       0000
                                   00000000     .PSECT  Y$EXEPAGED
                                       0000
             4B 30 5B 1B 41 31 5B 1B   0000     erase_pat:      .ascii  /[1A[0K/
                                       0008     assume .-erase_pat EQ 8                         ; so quadword access can be done
                                       0008
55 21 5B 1B 37 1B 00000010'010E0000'   0008     screen_ctrstr:  .ascid  /7[!UB;1H[K!AD!AD8/
41 21 44 41 21 4B 5B 1B 48 31 3B 42    0016
                            38 1B 44   0022
                                       0025
```

```
                   0025          .SBTTL  EXE$BRKTHRU - Break though write
                   0025
                   0025  ;++
                   0025  ;
                   0025  ; FUNCTIONAL DESCRIPTION:
                   0025  ;
                   0025  ;
                   0025  ; CALLING SEQUENCE:
                   0025  ;         NONE
                   0025  ;
                   0025  ; INPUT PARAMETERS:
                   0025  ;
                   0025  ;         R4 - PCB
                   0025  ;         AP - argument list
                   0025  ;
                   0025  ; IMPLICIT INPUTS:
                   0025  ;         NONE
                   0025  ;
                   0025  ; OUTPUT PARAMETERS:
                   0025  ;         NONE
                   0025  ;
                   0025  ; IMPLICIT OUTPUTS:
                   0025  ;         NONE
                   0025  ;
                   0025  ; COMPLETION CODES:
                   0025  ;         NONE
                   0025  ;
                   0025  ; SIDE EFFECTS:
                   0025  ;         NONE
                   0025  ;
                   0025  ;--
                   0025
            0FFC   0025          .ENTRY  EXE$BRKTHRU,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                   0027
                   0027  ;
                   0027  ; Check parameters and do initialization needed
                   0027  ;
        56    D4   0027          CLRL    R6                      ; no buffer yet
                   0029  ;
                   0029  ; Clear Event Flag
                   0029  ;
  53  04 AC   9A   0029          MOVZBL  EFN(AP),R3              ; Fetch EFN
00000000'EF   16   002D          JSB     SCH$CLREF               ; Clear
     4F 50    E9   0033          BLBC    R0,20$                  ; Exit on error
                   0036  ;
                   0036  ; Verify IOSB and clear it
                   0036  ;
  5B  14 AC   D0   0036          MOVL    IOSB(AP),R11            ; Get address of IOSB
        0B    13   003A          BEQL    10$                     ; Branch if none
                   003C          IFWRT   #8,(R11),5$             ; Branch if ok
      009F    31   0042          BRW     ACCVIO_EXIT             ; Error if not writeable
        6B    7C   0045  5$:     CLRQ    (R11)                   ; Clear
                   0047  10$:
  51  08 AC   D0   0047          MOVL    MSGBUF(AP),R1           ; Message buffer descriptor
00000000'GF   16   004B          JSB     G^EXE$PROBER_DSC        ; Probe descriptor
     31 50    E9   0051          BLBC    R0,20$                  ; branch if error
                   0054  ;
```

```
                               0054       ; R1 and R2 have length and address, calculate size of buffer
                               0054       ; needed for storage.
                               0054       ;
             51    51    3C    0054       MOVZWL  R1,R1                       ; clear top word
             59    51    7D    0057       MOVQ    R1,R9                       ; save both
        53    8E 8F    9A      005A       MOVZBL  #BRK$C_LENGTH,R3            ; Size of basic block
             53    51    C0    005E       ADDL    R1,R3                       ; For normal data
58    51   000000D0 8F    C1   0061       ADDL3   #16+<8*BRK_C_MAXLINES>,R1,R8     ; screen overhead and message
             53    58    C0    0069       ADDL    R8,R3                       ; For screen data
             53    03    C0    006C       ADDL    #3,R3                       ; round of to longword by adding and...
             53    03    CA    006F       BICL    #3,R3                       ;         clearing bits
             57    53    D0    0072       MOVL    R3,R7                       ; Save this length
                   04    C5    0075       MULL3   #BRK_C_SIMULCAST,-
             50    0E          0077               #BRK2$C_LENGTH,R0           ; Size of context area
             53    50    C0    0079       ADDL    R0,R3                       ; add to length
                               007C       ;
                               007C       ; Compute pages and allocate region
                               007C       ;
             51    53    D0    007C       MOVL    R3,R1                       ; Number of bytes
       00000000'GF     16      007F       JSB     G^EXE$ALOP1IMAG             ; Allocate memory
             5F    50    E9    0085  20$: BLBC    R0,ERROR_EXIT               ; exit on error
                               0088       ;
                               0088       ; Copy remaining paramters into allocated region
                               0088       ;
             56    52    D0    0088       MOVL    R2,R6                       ; Copy Address of block
                   1E    BB    008B       PUSHR   #^M<R1,R2,R3,R4>            ; Save
       00    6E    00    2C    008D       MOVC5   #0,(SP),#0,-
       62  008E 8F             0091               #BRK$C_LENGTH,(R2)          ; Zero entire structure (up to text)
                   1E    BA    0095       POPR    #^M<R1,R2,R3,R4>            ; Restore
                               0097
       08 A6       51    B0    0097       MOVW    R1,BRK$W_SIZE(R6)           ; And size
    60 A6       6647     9E    009B       MOVAB   (R6)[R7],BRK$L_QIOCTX(R6)   ; Qio context start address
       68 A6       58    D0    00A0       MOVL    R8,BRK$L_SCRMSGLEN(R6)      ; init
       1C A6       54    D0    00A4       MOVL    R4,BRK$L_PCB(R6)            ; Save PCB
       20 A6       5B    D0    00A8       MOVL    R11,BRK$L_IOSB(R6)          ; Set address
                               00AC       ;
                               00AC       ; Copy main message buffer
                               00AC       ;
     008C C6       59    B0    00AC       MOVW    R9,BRK$W_MSGLEN(R6)         ; Save length
          6A       59    28    00B1       MOVC3   R9,(R10),-
     008E C6                   00B4               BRK$T_MSGBUF(R6)            ; Copy message buffer
       6C A6       53    D0    00B7       MOVL    R3,BRK$L_SCRMSG(R6)         ; next byte is where screen message starts
                               00BB       ;
                               00BB       ; Copy send type and "send to:" string (if required)
                               00BB       ;
            027B         30    00BB       BSBW    GET_SENDTO                  ; handle SENDTO, SENDTYPE
             26 50       E9    00BE       BLBC    R0,ERROR_EXIT               ; check status
                               00C1       ;
                               00C1       ; Set up time quadword if timeout requested
                               00C1       ;
       50    24 AC       D0    00C1       MOVL    TIMOUT(AP),R0               ; Timeout value
                   12    13    00C5       BEQL    240$                        ; branch if none specified
             50    04    D1    00C7       CMPL    #BRK_C_MINTIME,R0           ; Compare to minimum number of seconds
                   13    14    00CA       BGTR    BADPARAM_EXIT               ; Exit if too small
             50    50    CE    00CC       MNEGL   R0,R0                       ; Get negative value
00    50   00989680 8F    7A   00CF       EMUL    #10*1000*1000,R0,#0,-
                2C A6          00D7               BRK$Q_TIMEOUT(R6)           ; Times ten million ticks per second
            0F    B0  00D9  240$:    MOVW    #BRK_C_CLUTIMEOUT,-
         4E A6        00DB                   BRK$W_SECONDS(R6)       ; set default timeout for cluster
            10    11  00DD           BRB     ALL_OK                  ; And continue
                      00DF
                      00DF           ;
                      00DF           ; An error has occured in initial processing...
                      00DF           ;
                      00DF  BADPARAM_EXIT:
       50   14    3C  00DF           MOVZWL  #SS$_BADPARAM,R0        ; set status
            03    11  00E2           BRB     ERROR_EXIT              ; exit
                      00E4  ACCVIO_EXIT:
       50   0C    3C  00E4           MOVZWL  #SS$_ACCVIO,R0          ; Set error
                      00E7  ERROR_EXIT:
            56    D5  00E7           TSTL    R6                      ; Buffer to delete?
            03    13  00E9           BEQL    10$                     ; Branch if not
          056E    30  00EB           BSBW    RETURN_MEMORY           ; return memory
                      00EE  10$:
                  04  00EE           RET                             ; exit
                      00EF
                      00EF           ;
                      00EF           ; Copy remaining parameters...
                      00EF           ;
                      00EF  ALL_OK:
50    A0000000 8F   D0  00EF         MOVL    #<PRV$M_BYPASS!PRV$M_SHARE>,R0  ; privileges required
                      00F6           ASSUME  PHD$Q_PRIVMSK EQ 0              ; for indirection
      54    1C A6   D0  00F6         MOVL    BRK$L_PCB(R6),R4                ; Set PCB address
66    50    6C B4   CB  00FA         BICL3   @PCB$L_PHD(R4),R0,BRK$Q_PRIVS(R6) ; Clear those already set
                      00FF
                      00FF           ASSUME  BRK$W_EFN+2 EQ BRK$B_STS        ; assumes so next instruction
                      00FF           ASSUME  BRK$W_EFN+3 EQ BRK$B_PRVMODE    ; can set efn and zero sts and prvmo
   64 A6    04 AC   3C  00FF         MOVZWL  EFN(AP),BRK$W_EFN(R6)           ; Copy event flag number
      50    20 AC   D0  0104         MOVL    REQID(AP),R0                    ; Requestor ID
       50      3F   D1  0108         CMPL    #63,R0                          ; Check legal (0-63 legal)
            D2 1F       010B         BLSSU   BADPARAM_EXIT                   ; exit if not
      50 A6    50   D0  010D         MOVL    R0,BRK$L_REQID(R6)              ; Save Requestor ID
   38 A6    1C AC   D0  0111         MOVL    FLAGS(AP),BRK$L_FLAGS(R6)       ; Flags
   34 A6    18 AC   D0  0116         MOVL    CARCON(AP),BRK$L_CARCON(R6)     ; Set carriage control
   24 A6    28 AC   D0  011B         MOVL    ASTADR(AP),BRK$L_ASTADR(R6)     ; Ast routine
   28 A6    2C AC   D0  0120         MOVL    ASTPRM(AP),BRK$L_ASTPRM(R6)     ; Ast routine parameter
                      0125           ;
                      0125           ; Other misc. initialization
                      0125           ;
                      0125           ASSUME  BRK$W_STATUS+2 EQ BRK$W_SUCCESSCNT
                      0125           ASSUME  BRK$W_STATUS+4 EQ BRK$W_TIMEOUTCNT
                      0125           ASSUME  BRK$W_STATUS+6 EQ BRK$W_REFUSEDCNT
      70 A6    01   9B  0125         MOVZBW  #SS$_NORMAL,BRK$W_STATUS(R6)    ; Assume final status
   78 A6   0000'8F  B0  0129         MOVW    #MSG$_TRMBRDCST,BRK$W_TRMMSG(R6); set mailbox prefix code
                      012F           ;
                      012F           ; read PSL and save previous mode
                      012F           ;
            50    DC  012F           MOVPSL  R0                              ; fetch PSL
       02   16    EF  0131           EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,-
       50   50        0134                   R0,R0                           ; extract previous mode
      67 A6    50   90  0136         MOVB    R0,BRK$B_PRVMODE(R6)            ; save
                      013A           ;
                      013A           ; Set up search contexts
                      013A           ;
       54 A6    01   CE  013A          MNEGL   #1,BRK$L_PIDCTX(R6)            ; wild card pid
                         013E          ASSUME  BRK$L_UCBCTX+4 EQ BRK$L_DDBCTX  ; assume alignment
                         013E
                         013E          ;
                         013E          ; Format screen message (if SCREEN requested)
                         013E          ;
       57    38 A6   D0  013E          MOVL    BRK$L_FLAGS(R6),R7          ; Flags parameter
     4D 57    08     E1  0142          BBC     #BRK$V_SCREEN,R7,100$       ; Skip if not requested
       50    57      9A  0146          MOVZBL  R7,R0                       ; lines to clear
       50    18      D1  0149          CMPL    #BRK_C_MAXLINES,R0          ; Greater than max?
             91      1F  014C          BLSSU   BADPARAM_EXIT               ; Branch if yes
       51    50      D0  014E          MOVL    R0,R1                       ; copy
    52 51    08      C5  0151          MULL3   #8,R1,R2                    ; bytes of erase pattern
                         0155          ;
                         0155          ; Set up repeating erase line pattern on stack
                         0155          ;
  7E    FEA7 CF      7D  0155  10$:    MOVQ    W^ERASE_PAT,-(SP)           ; copy erase pattern
          F8 50      F5  015A          SOBGTR  R0,10$                      ; one for each line
       53    5E      D0  015D          MOVL    SP,R3                       ; address of erase pattern
     04 57    09     E1  0160          BBC     #BRK$V_BOTTOM,R7,20$        ; Branch if message on top of screen
       51    84 8F   9A  0164          MOVZBL  #132,R1                     ; Set "bottom" (note 132 >> 24)
                         0168  20$:
  54    008C C6      3C  0168          MOVZWL  BRK$W_MSGLEN(R6),R4         ; Size
  55    008E C6      9E  016D          MOVAB   BRK$T_MSGBUF(R6),R5         ; address of data
                         0172          $FAO_S  CTRSTR = SCREEN_CTRSTR,-
                         0172                  OUTLEN = BRK$L_SCRMSGLEN(R6),-
                         0172                  OUTBUF = BRK$L_SCRMSGLEN(R6),-
                         0172                  P1 = R1,-                   ; position top/bottom
                         0172                  P2 = R2,-                   ; lines to erase * 8
                         0172                  P3 = R3,-                   ; erase pattern address
                         0172                  P4 = R4,-                   ; size of msgbuf
                         0172                  P5 = R5                     ; msgbuf address
          03 50      E8  018D          BLBS    R0,100$
           FF54      31  0190          BRW     ERROR_EXIT                  ; blew it
                         0193  100$:
                         0193          ;
                         0193          ; Start initial QIO's up. AST's are disabled first so that a
                         0193          ; CPU limit exceeded ast cannot fire between assigning the
                         0193          ; channel and setting the CCB$M_IMGTMP flag. Something that would cause
                         0193          ; image exit to occur before the IMGTMP flag was set cannot be allowed.
                         0193          ; Disabling AST makes synchronization of CHECK_COMPLETE easier as well.
                         0193          ;
                         0193          $SETAST_S ENBFLG = #0               ; Disable AST's
                         019C          ;
                         019C          ;(At this point, R6 points to BRK structure, all others are scratch)
                         019C          ;
       57    60 A6   D0  019C          MOVL    BRK$L_QIOCTX(R6),R7         ; QIO context area
       58    04      3C  01A0          MOVZWL  #BRK_C_SIMULCAST,R8         ; Number to do at one time
                         01A3  300$:
       67    56      D0  01A3          MOVL    R6,BRK2$L_COMMON(R7)        ; Point back to common region
             4F      10  01A6          BSBB    DO_WRITE                    ; Do the write
          07 50      E9  01A8          BLBC    R0,350$                     ; exit on error
       57    0E A7   9E  01AB          MOVAB   BRK2$C_LENGTH(R7),R7        ; Add size to qio context
          F1 58      F5  01AF          SOBGTR  R8,300$                     ; Continue
                         01B2  350$:
             50      DD  01B2          PUSHL   R0                          ; Save status
                         01B4          ;
                         01B4          ; Before returning to user, see if there is a cluster to send to
```

```
                               01B4           ;
                    0B    E1   01B4           BBC     #BRK$V_CLUSTER,-               ;
              0E 38 A6         01B6                   BRK$L_FLAGS(R6),360$           ; Branch if "cluster" not requested
                               01B9           IFNOCLSTR 360$                         ; or if not in cluster
         00000000'GF      16   01C1           JSB     G^EXE$CSP_BRKTHRU              ; send message
                               01C7     360$:
                  044E    30   01C7           BSBW    CHECK_COMPLETE                 ; done? Deallocate BRK if so
                               01CA           $SETAST_S ENBFLG = #1                  ; Enable AST's
                    50  8ED0   01D3           POPL    R0                             ; Restore status
       50      2894 8F    B1   01D6           CMPW    #SS$_NOOPER,R0                 ; no OPER priv?
                    03    12   01DB           BNEQ    365$                           ; continue if not
                  FF07    31   01DD           BRW     ERROR_EXIT                     ; take error exit
                               01E0     365$:
             50     01    9A   01E0           MOVZBL  #SS$_NORMAL,R0                 ; Set success for everything else
                          04   01E3     370$: RET                                    ; Return to user
                               01E4
                          01E4        .SBTTL  DO_WRITE - Queue a single write request
                          01E4  ;++
                          01E4  ;
                          01E4  ; FUNCTIONAL DESCRIPTION:
                          01E4  ;
                          01E4  ;
                          01E4  ; CALLING SEQUENCE:
                          01E4  ;       BSBW    DO_WRITE
                          01E4  ;
                          01E4  ; INPUT PARAMETERS:
                          01E4  ;
                          01E4  ;       R6 - BRK
                          01E4  ;       R7 - QIO context area
                          01E4  ;
                          01E4  ; IMPLICIT INPUTS:
                          01E4  ;       NONE
                          01E4  ;
                          01E4  ; OUTPUT PARAMETERS:
                          01E4  ;       NONE
                          01E4  ;
                          01E4  ; IMPLICIT OUTPUTS:
                          01E4  ;       NONE
                          01E4  ;
                          01E4  ; COMPLETION CODES:
                          01E4  ;       R0 - status
                          01E4  ;
                          01E4  ;               SS$_NORMAL - all ok or error set in STATUS
                          01E4  ;               SS$_NOMOREPROC - done with all QIO's
                          01E4  ;
                          01E4  ; SIDE EFFECTS:
                          01E4  ;
                          01E4  ;       Destroys R1,R2,R3,R4,R5
                          01E4  ;
                          01E4  ;--
                          01E4
                          01E4  UNLOCK_DB:
              00    E5    01E4          BBCC    #BRK$V_LOCKED,-
        0D 66 A6          01E6                  BRK$B_STS(R6),10$       ; clear locked flag
     54    1C A6    D0    01E9          MOVL    BRK$L_PCB(R6),R4        ; PCB
    00000000'GF     16    01ED          JSB     G^SCH$IOUNLOCK          ; unlock
                          01F3          SETIPL  #0                      ; lower IPL
                    05    01F6  10$:    RSB                             ; Return
                          01F7
                          01F7  DO_WRITE:
                          01F7
                          01F7  10$:
              EB    10    01F7          BSBB    UNLOCK_DB               ; Unlock data base
            01FA    30    01F9          BSBW    GET_NEXT_TERMINAL       ; Get next terminal
                          01FC          ;
                          01FC          ; returns with I/O database locked at IPL 2
                          01FC          ;
           E5 50    E9    01FC          BLBC    R0,UNLOCK_DB    ; branch if done (no more processes)
                          01FF          ;
                          01FF          ; Test for broadcast to mailbox
                          01FF          ;
     55    58 A6    D0    01FF          MOVL    BRK$L_UCBCTX(R6),R5     ; fetch UCB address
              04    E1    0203          BBC     #TT2$V_BRDCSTMBX,-
            23 48 A5      0205                    UCB$L_DEVDEPND2(R5),40$ ; Branch if not allowed
               55   DD    0208          PUSHL     R5                      ; Save ucb address
         55 60 A5   D0    020A          MOVL      UCB$L_AMB(R5),R5        ; Get address of associated mailbox
               18   13    020E          BEQL      30$                     ; Branch if none
                          0210          ;
                          0210          ; Send broadcast to assoicated mailbox
                          0210          ;
       53 008C C6   3C    0210          MOVZWL    BRK$W_MSGLEN(R6),R3     ; Get length of message
            53 16   C0    0215          ADDL2     #<BRK$T_MSGBUF-BRK$W_TRMMSG>,R3 ; Add mailbox prefix overhead
         54 78 A6   9E    0218          MOVAB     BRK$W_TRMMSG(R6),R4     ; Set address of mailbox message
       00000000'GF  16    021C          JSB       G^EXE$WRTMAILBOX        ; Send message
            03 50   E9    0222          BLBC      R0,30$                  ; branch if error sending to mailbox
            72 A6   B6    0225          INCW      BRK$W_SUCCESSCNT(R6)    ; One more successful completion
                          0228  30$:
            55 8ED0       0228          POPL      R5                      ; Restore ucb address
                          022B  40$:
       00020001 8F  D3    022B          BITL      #<TT$M_NOBRDCST!TT$M_PASSALL>,-
            44 A5         0231                    UCB$L_DEVDEPEND(R5)     ; test for NOBROADCAST or PASSALL
               C2   12    0233          BNEQ      10$                     ; skip if either set
               AD   10    0235          BSBB      UNLOCK_DB               ; unlock data base
                          0237          ;
                          0237          ; Assign channel (if possible)
                          0237          ;
               66   D5    0237          TSTL      BRK$Q_PRIVS(R6)         ; assumes no privs in high longword
               0F   13    0239          BEQL      42$                     ; privs required non-null
                          023B          $SETPRV_S -                       ;
                          023B                    ENBFLG = #1,-                   ; Enable privs
                          023B                    PRVADR = BRK$Q_PRIVS(R6)        ; Privs to set
                          024A  42$:
         52 7E      7E    024A          MOVAQ     -(SP),R2                ; Allocate descriptor on stack
      62 0C A6      9A    024D          MOVZBL    BRK$T_DEVNAME(R6),(R2)  ; Length
   04 A2 0D A6      9E    0251          MOVAB     BRK$T_DEVNAME+1(R6),4(R2) ; address
                          0256
                          0256          $ASSIGN_S -
                          0256                    DEVNAM = (R2),-         ; device name
                          0256                    CHAN = BRK2$W_CHAN(R7)  ; channel
         5E 08      C0    0264          ADDL      #8,SP                   ; pop descriptor
            19 50   E8    0267          BLBS      R0,50$                  ; branch if ok
            76 A6   B6    026A          INCW      BRK$W_REFUSEDCNT(R6)    ; Refused
      70 A6 50      B0    026D  45$:    MOVW      R0,BRK$W_STATUS(R6)     ; record status
                          0271          $SETPRV_S -                       ;
                          0271                    ENBFLG = #0,-                   ; Disable privs
                          0271                    PRVADR = BRK$Q_PRIVS(R6)        ; Privs to disable
             FF74   31    0280          BRW       10$                     ; Try another terminal
                          0283          ;
                          0283          ; modify the CCB so that the channel will be run down at image exit
                          0283          ;
                          0283  50$:
                          0283          $SETPRV_S -                       ;
                          0283                    ENBFLG = #0,-                   ; Disable privs
                          0283                    PRVADR = BRK$Q_PRIVS(R6)        ; Privs to reset
                          0292
      50 0C A7      3C    0292          MOVZWL    BRK2$W_CHAN(R7),R0      ; Channel number
         50 50      CE    0296          MNEGL     R0,R0                   ; Get negative
50  00000000'FF40   9E    0299          MOVAB     @CTL$GL_CCBBASE[R0],R0  ; Get CCB address
               02   88    02A1          BISB      #CCB$M_IMGTMP,-
            08 A0         02A3                    CCB$B_STS(R0)           ; Set image temporary channel
```

```
                             02A5            ;
                             02A5            ; Do QIO
                             02A5            ;
 51   008E C6    9E          02A5            MOVAB   BRK$T_MSGBUF(R6),R1          ; assume standard message
 52   008C C6    3C          02AA            MOVZWL  BRK$W_MSGLEN(R6),R2          ; and length
   53   34 A6    D0          02AF            MOVL    BRK$L_CARCON(R6),R3          ; and carriage control
 54   2270 8F    3C          02B3            MOVZWL  #<IO$_WRITEVBLK!-
                             02B8                    IO$M_REFRESH!-
                             02B8                    IO$M_BREAKTHRU!-
                             02B8                    IO$M_CANCTRLO>,R4            ; I/O function code
           08    E1          02B8            BBC     #BRK$V_SCREEN,-
      11 38 A6               02BA                    BRK$L_FLAGS(R6),70$          ; Branch if screen not requested
           1D    E1          02BD            BBC     #TT2$V_DECCRT,-
      0C 48 A5               02BF                    UCB$L_DEVDEPND2(R5),70$      ; or not dec crt
 51     6C A6    D0          02C2            MOVL    BRK$L_SCRMSG(R6),R1          ; screen message
 52     68 A6    3C          02C6            MOVZWL  BRK$L_SCRMSGLEN(R6),R2       ; and length
           53    D4          02CA            CLRL    R3                           ; no carriage control
           05    11          02CC            BRB     75$                          ; force no refresh for screen write
                             02CE     70$:
           0A    E1          02CE            BBC     #BRK$V_NOREFRESH,-
      05 38 A6               02D0                    BRK$L_FLAGS(R6),77$          ; Branch if not NO REFRESH
 54   2000 8F    AA          02D3     75$:   BICW    #IO$M_REFRESH,R4             ; Clear refresh flag
                             02D8     77$:
                             02D8            ;
                             02D8            ; Do the QIO!
                             02D8            ;
                             02D8            $QIO_S  CHAN = BRK2$W_CHAN(R7),-
                             02D8                    EFN  = #BRK_C_QIOEFN,-
                             02D8                    FUNC = R4,-
                             02D8                    IOSB = BRK2$Q_IOSB(R7),-
                             02D8                    ASTADR = QIO_DONE,-
                             02D8                    ASTPRM = R7,-                ; qio context
                             02D8                    P1 = (R1),-                  ; address
                             02D8                    P2 = R2,-                    ; and length
                             02D8                    P4 = R3                      ; Carriage control
        27 50    E9          02FD            BLBC    R0,200$                      ; error from QIO?
        0A A6    B6          0300            INCW    BRK$W_OUTCNT(R6)             ; Increment outstanding count
                             0303            ;
                             0303            ; Set timer for timeout if requested
                             0303            ;
        2C A6    7D          0303            MOVQ    BRK$Q_TIMEOUT(R6),-          ; (Test quad)
        2C A6                0306                    BRK$Q_TIMEOUT(R6)            ; Time out requested?
           19    13          0308            BEQL    80$                          ; Branch if not
                             030A
                             030A            $SETIMR_S -
                             030A                    EFN    = #BRK_C_TIMEFN, -
                             030A                    DAYTIM = BRK$Q_TIMEOUT(R6), -
                             030A                    ASTADR = W^QIO_TIMEOUT, -
                             030A                    REQIDT = R7
                             031C
        04 50    E8          031C            BLBS    R0,80$                       ; branch if ok
   70 A6   50    B0          031F            MOVW    R0,BRK$W_STATUS(R6)          ; Set final status
                             0323     80$:
      50   01    9A          0323            MOVZBL  #SS$_NORMAL,R0               ; exit
                             0326     100$:
           05                0326            RSB
                             0327     ;
```

```
                        0327  ; Error during QIO
                        0327  ;
                        0327  200$:
     70 A6    50    B0  0327          MOVW    R0,BRK$W_STATUS(R6)      ; Set final status
                        032B          $DASSGN_S CHAN = BRK2$W_CHAN(R7)  ; Deassign channel
          FEBE    31    0336          BRW     10$                      ; Try again with this QIO context
                        0339
                        0339          .SBTTL  GET_SENDTO - Handle SENDTO and SENDTYPE inputs
                        0339     ;++
                        0339     ;
                        0339     ; FUNCTIONAL DESCRIPTION:
                        0339     ;
                        0339     ;        Handle the SENDTYPE and SENDTO parameters and set up BRK.
                        0339     ;        Privilege is checked for all but BRK$C_DEVICE writes.
                        0339     ;        Writes to same username are allowed without privilege.
                        0339     ;
                        0339     ; CALLING SEQUENCE:
                        0339     ;
                        0339     ;        BSBW    GET_SENDTO
                        0339     ;
                        0339     ; INPUT PARAMETERS:
                        0339     ;
                        0339     ;        R6 - BRK
                        0339     ;        SENDTYPE(AP) - sendtype parameter
                        0339     ;        SENDTO(AP)   - sendto parameter
                        0339     ;
                        0339     ; IMPLICIT INPUTS:
                        0339     ;        NONE
                        0339     ;
                        0339     ; OUTPUT PARAMETERS:
                        0339     ;        NONE
                        0339     ;
                        0339     ; IMPLICIT OUTPUTS:
                        0339     ;        NONE
                        0339     ;
                        0339     ; COMPLETION CODES:
                        0339     ;
                        0339     ;        R0 - success or failure
                        0339     ;
                        0339     ; SIDE EFFECTS:
                        0339     ;
                        0339     ;        R1-R5,R7 are destroyed.
                        0339     ;--
                        0339
                        0339     GET_SENDTO:
                        0339
57   10 AC       D0     0339             MOVL    SENDTYPE(AP),R7          ; fetch Send type
   57    04      D1     033D             CMPL    #BRK$C_MAXSENDTYPE,R7    ; Compare to maximum
         12      1F     0340             BLSSU   5$                       ; branch if error
                        0342
4C A6    57      B0     0342             MOVW    R7,BRK$W_SENDTYPE(R6)    ; Save low order word
                        0346             CASE    R7,-                     ; Case on send type
                        0346                     <5$,-                    ; Invalid
                        0346                     10$,-                    ; send to device name
                        0346                     10$,-                    ; send to username
                        0346                     150$,-                   ; send to all users
                        0346                     150$>,-                  ; send to all terminals
                        0346                     TYPE = W                 ; word context
   50    14      3C     0354     5$:     MOVZWL  #SS$_BADPARAM,R0         ; Set status
                 05     0357     7$:     RSB
                        0358             ;
                        0358             ; single device or username requested
                        0358             ;
51   0C AC       D0     0358     10$:    MOVL    SENDTO(AP),R1            ; Get "send to" address
```

```
   00000000'GF    16  035C          JSB     G^EXE$PROBER_DSC        ; test for read
          F2 50   E9  0362          BLBC    R0,7$                   ; exit on error
       51    51   3C  0365          MOVZWL  R1,R1                   ; zero high word
             EA   13  0368          BEQL    5$                      ; Must be non-zero
                      036A
       57    01   91  036A          CMPB    #BRK$C_DEVICE,R7        ; device
             28   13  036D          BEQL    40$                     ; Branch if yes
                      036F  ;
                      036F  ; Must be Username
                      036F  ;
       51    0C   B1  036F          CMPW    #JIB$S_USERNAME,R1      ; max user name length
             E0   1F  0372          BLSSU   5$                      ; error if so
    3C A6    51   90  0374          MOVB    R1,BRK$T_SENDNAME(R6)   ; simply copy username ascic string
             51   DD  0378          PUSHL   R1                      ; Save Length
       62    51   28  037A          MOVC3   R1,(R2),-
          3D A6       037D                  BRK$T_SENDNAME+1(R6)    ; and copy string
          51 8ED0     037F          POPL    R1                      ; Restore Length
    54    1C A6   D0  0382          MOVL    BRK$L_PCB(R6),R4        ; Fetch PCB address
 54    0080 C4    D0  0386          MOVL    PCB$L_JIB(R4),R4        ; Fetch JIB
                      038B  ;
                      038B  ; JIB$T_USERNAME is a 12 byte field, with NO BYTE COUNT!
                      038B  ;
          0C      2D  038B          CMPC5   #JIB$S_USERNAME,-
    20    0C A4       038D                  JIB$T_USERNAME(R4),#^A/ /,-
    3D A6    51       0390                  R1,BRK$T_SENDNAME+1(R6) ; compare strings, fill with blanks
             51   12  0393          BNEQ    150$                    ; branch if not equal
             4B   11  0395          BRB     50$                     ; names are same, no priv required
                      0397  ;
                      0397  ; Device name, do a GETDVI to translate logical name
                      0397  ;
                      0397  40$:
       54    5E   D0  0397          MOVL    SP,R4                   ; Save SP
       55    7E   DE  039A          MOVAL   -(SP),R5                ; allocate scratch longword
             7E   D4  039D          CLRL    -(SP)                   ; end of list
             55   DD  039F          PUSHL   R5                      ; just a longword for device name length
          0D A6   9F  03A1          PUSHAB  BRK$T_DEVNAME+1(R6)     ; copy directly into device name area
   0020000F 8F    DD  03A4          PUSHL   #<DVI$_DEVNAM@16>!-
                      03AA                  <BRK$S_DEVNAME-1>       ; size and getdvi code
       53    5E   D0  03AA          MOVL    SP,R3                   ; save
             52   DD  03AD          PUSHL   R2                      ; address (device descriptor)
             51   DD  03AF          PUSHL   R1                      ; length
       51    5E   D0  03B1          MOVL    SP,R1                   ; save
                      03B4          $GETDVIW_S -
                      03B4                  EFN = #BRK_C_DVIEFN,-   ; event flag number
                      03B4                  DEVNAM = (R1),-         ; get device name (and wait)
                      03B4                  ITMLST = (R3)           ; Item list
    0C A6    65   90  03CA          MOVB    (R5),BRK$T_DEVNAME(R6)  ; Copy length
       5E    54   D0  03CE          MOVL    R4,SP                   ; Restore SP
       0C A6      7D  03D1          MOVQ    BRK$T_DEVNAME(R6),-
       3C A6          03D4                  BRK$T_SENDNAME(R6)      ; copy in case of cluster broadcast
       14 A6      7D  03D6          MOVQ    BRK$T_DEVNAME+8(R6),-
       44 A6          03D9                  BRK$T_SENDNAME+8(R6)    ; copy in case of cluster broadcast
       07 50      E9  03DB          BLBC    R0,110$                 ; check status
          04      88  03DE          BISB    #BRK$M_CHKPRIV,-
       66 A6          03E0                  BRK$B_STS(R6)           ; Set "check priv later" bit
                      03E2  50$:
       50    01   3C  03E2          MOVZWL  #SS$_NORMAL,R0          ; set ok
```

```
                             05  03E5    758 110$:   RSB
                                 03E6    759         ;
                                 03E6    760         ;  Check for OPER priv before allowing request
                                 03E6    761         ;
         54    1C A6         D0  03E6    762 150$:   MOVL    BRK$L_PCB(R6),R4          ; Fetch PCB address
                                 03EA    763         IFPRIV  OPER,50$                  ; If priv ok, continue
      50    2894 8F          3C  03F0    764         MOVZWL  #SS$_NOOPER,R0            ; Set status
                             05  03F5    765         RSB                               ; exit
                                 03F6    766
```

```
                              03F6          .SBTTL  GET_NEXT_TERMINAL - return next terminal
                              03F6   ;++
                              03F6   ;
                              03F6   ; FUNCTIONAL DESCRIPTION:
                              03F6   ;
                              03F6   ;       Given context in BRK, determine next terminal to send message to.
                              03F6   ;
                              03F6   ; CALLING SEQUENCE:
                              03F6   ;
                              03F6   ;       BSBW    GET_NEXT_TERMINAL
                              03F6   ;
                              03F6   ; INPUT PARAMETERS:
                              03F6   ;
                              03F6   ;       R6 - BRK
                              03F6   ;       R7 - QIO context
                              03F6   ;
                              03F6   ; IMPLICIT INPUTS:
                              03F6   ;       NONE
                              03F6   ;
                              03F6   ; OUTPUT PARAMETERS:
                              03F6   ;       NONE
                              03F6   ;
                              03F6   ; IMPLICIT OUTPUTS:
                              03F6   ;
                              03F6   ;       If R0 = success, then BRK$T_DEVNAME is filled in,
                              03F6   ;                         and BRK$L_UCBCTX has UCB address.
                              03F6   ;
                              03F6   ; COMPLETION CODES:
                              03F6   ;
                              03F6   ;       R0 -    SS$_NORMAL
                              03F6   ;               SS$_NOMOREPROC
                              03F6   ;       other errors returned in BRK$W_STATUS
                              03F6   ;
                              03F6   ; SIDE EFFECTS:
                              03F6   ;
                              03F6   ;       Destroys R1,R2,R3,R4,R5
                              03F6   ;
                              03F6   ;--
                              03F6
                              03F6   GET_NEXT_TERMINAL:
                              03F6
 50    09A8 8F        3C      03F6          MOVZWL  #SS$_NOMOREPROC,R0      ; assume no more processes to send to
            01        E1      03FB          BBC     #BRK$V_DONE,-
      01 66 A6                03FD                  BRK$B_STS(R6),5$        ; If not done, lookup next terminal
                      05      0400          RSB                             ; Return all done once again
                              0401   5$:
                              0401          CASE    BRK$W_SENDTYPE(R6),-    ; Case on send type
                              0401                  <10$,-                  ; Invalid
                              0401                  100$,-                  ; send to device name
                              0401                  200$,-                  ; send to username
                              0401                  ALL_TERMS,-             ; send to all users
                              0401                  ALL_TERMS>,-            ; send to all terminals
                              0401                  TYPE = W                ; word context
                              0410
      50   14         3C      0410   10$:   MOVZWL  #SS$_BADPARAM,R0        ; bad parameter
         0085         31      0413          BRW     NEXT_TERM_ERROR         ; error
                              0416
```

```
                                 0416            ;
                                 0416            ; Send to one device
                                 0416            ;
                                 0416      100$:
          66 A6    02     88     0416            BISB    #BRK$M_DONE,BRK$B_STS(R6) ; set done
                 008C     31     041A            BRW     HAVE_NAME                 ; and go
                                 041D            ;
                                 041D            ; map username into terminal name
                                 041D            ;
                                 041D      200$:
             5E    2C     C2     041D            SUBL2   #STK$C_LEN,SP             ; Allocate some work space
             52    5E     D0     0420            MOVL    SP,R2                     ; copy pointer
                                 0423            ;
                                 0423            ; Initialize area for GETJPI call
                                 0423            ;
                                 0423      210$:
             51    52     D0     0423            MOVL    R2,R1                     ; copy pointer
                                 0426
     B1   0202000C 8F     D0     0426            MOVL    #<JPI$_USERNAME@16>!-
                                 042D                    <JIB$S_USERNAME>,(R1)+  ; username size and code
          81    1E A2     9E     042D            MOVAB   STK$T_USERNAME(R2),(R1)+; username address
          81    1C A2     9E     0431            MOVAB   STK$W_USERLEN(R2),(R1)+ ; username length to return
                                 0435
     B1   031D000F 8F     D0     0435            MOVL    #<JPI$_TERMINAL@16>!-
                                 043C                    <BRK$S_DEVNAME-1>,(R1)+ ; terminal name size
          81    0D A6     9E     043C            MOVAB   BRK$T_DEVNAME+1(R6),(R1)+; terminal name address
          81    2A A2     9E     0440            MOVAB   STK$W_TERMLEN(R2),(R1)+ ; terminal name length to return
                   61     D4     0444            CLRL    (R1)                    ; End of list
                                 0446
                                 0446            $GETJPI_S -
                                 0446                    EFN    = #BRK_C_JPIEFN,-       ; efn
                                 0446                    PIDADR = BRK$L_PIDCTX(R6),-    ; pid context
                                 0446                    ITMLST = (R2)                  ; item list
             11    50     E8     045A            BLBS    R0,220$                 ; Branch if ok
          50       24     B1     045D            CMPW    #SS$_NOPRIV,R0          ; no priv ?
                   C1     13     0460            BEQL    210$                    ; yes, try again
                                 0462
             5E    2C     C0     0462            ADDL2   #STK$C_LEN,SP           ; Deallocate work space
        50       09A8 8F  B1     0465            CMPW    #SS$_NOMOREPROC,R0      ; no more processes?
                   38     13     046A            BEQL    NO_MORE_TERM            ; yes, done
                   2D     11     046C            BRB     NEXT_TERM_ERROR         ; No, unexpected error
                                 046E      220$:
             2A    A2     B5     046E            TSTW    STK$W_TERMLEN(R2)       ; Interactive?
                   B0     13     0471            BEQL    210$                    ; If zero, no, try again
                                 0473
                   0C     BB     0473            PUSHR   #^M<R2,R3>              ; Save
          50    3C A6     9A     0475            MOVZBL  BRK$T_SENDNAME(R6),R0   ; length
                1C A2     2D     0479            CMPC5   STK$W_USERLEN(R2),-     ; length
                1E A2            047C                    STK$T_USERNAME(R2),-    ; address of name returned
    3D A6    50    20            047E                    #^A/ /,R0,-             ; fill and length
                                 0482                    BRK$T_SENDNAME+1(R6)    ; requested name
                   0C     BA     0482            POPR    #^M<R2,R3>              ; restore, (does not affect CC)
                   9D     12     0484            BNEQ    210$                    ; not equal, loop
             2A    A2     90     0486            MOVB    STK$W_TERMLEN(R2),-
             0C    A6            0489                    BRK$T_DEVNAME(R6)       ; Length
             5E    2C     C0     048B            ADDL2   #STK$C_LEN,SP           ; Deallocate work space
                                 048E            ;
```

```
SYS
Sym

JPI
LOC
MSC
MSC
NEX
NO_
PCE
PCE
PCE
PCE
PCE
PHD
PR$
PRV
PRV
PRV
PRV
PRV
PSL
PSL
QIO
QIO
REC
RET
SCH
SCH
SCH
SCH
SCR
SEN
SEN
SS$
SS$
SS$
SS$
SS$
SS$
SS$
SS$
SS$
STK
STK
STK
STK
STK
STK
STK
STK
STK
STK
STK
STK
STK
STK
SYS
SYS
SYS
SYS
```

```
                     048E            ; Username match found, scan device name for unit number
                     048E            ;
             19  11  048E            BRB     HAVE_NAME                ; exit
                     0490            ;
                     0490            ; Send to all terminals/users
                     0490            ;
                     0490   ALL_TERMS:
           00DE  30  0490            BSBW    LOCKDB                   ; lock database
           00EB  30  0493            BSBW    FIND_NEXT_TERM           ; Find next terminal
          30 50  E8  0496            BLBS    R0,HAVE_UCB              ; Continue if OK
             04  11  0499            BRB     TERM_DONE                ; Return proper status
                     049B
                     049B   NEXT_TERM_ERROR:
      70 A6   50  B0  049B           MOVW    R0,BRK$W_STATUS(R6)      ; Set final status
                     049F
                     049F   TERM_DONE:
   50   09A8 8F  3C  049F            MOVZWL  #SS$_NOMOREPROC,R0       ; no more processes to send to
                     04A4
                     04A4   NO_MORE_TERM:
      66 A6   02  88  04A4           BISB    #BRK$M_DONE,BRK$B_STS(R6) ; set done
                  05  04A8           RSB
                     04A9
                     04A9   HAVE_NAME:
                     04A9
           00C5  30  04A9            BSBW    LOCKDB                   ; lock database
                     04AC            ;
                     04AC            ; Map name into UCB address of this terminal
                     04AC            ;
         0D A6   9F  04AC            PUSHAB  BRK$T_DEVNAME+1(R6)      ; address of device name
    7E   0C A6   9A  04AF            MOVZBL  BRK$T_DEVNAME(R6),-(SP)  ; Length
       51    5E  D0  04B3            MOVL    SP,R1                    ; Address of descriptor
    54   1C A6   D0  04B6            MOVL    BRK$L_PCB(R6),R4         ; Set PCB address
                     04BA
  00000000'GF    16  04BA            JSB     G^IOC$SEARCHDEV          ; find the UCB (puts addr in R1)
       5E    08  C0  04C0            ADDL    #8,SP                    ; pop descriptor
       D5    50  E9  04C3            BLBC    R0,NEXT_TERM_ERROR       ; error
       55    51  D0  04C6            MOVL    R1,R5                    ; UCB address
                     04C9
                     04C9   HAVE_UCB:
                     04C9            ;
                     04C9            ; Check availability, access and privilege
                     04C9            ;
             02  E1  04C9            BBC     #DEV$V_TRM,-
       28 38 A5      04CB                    UCB$L_DEVCHAR(R5),3$     ; skip if not terminal
             12  E1  04CE            BBC     #DEV$V_AVL,-
       23 38 A5      04D0                    UCB$L_DEVCHAR(R5),3$     ; skip terminal if not available
        2040 8F  B3  04D3            BITW    #<DEV$M_NET!DEV$M_SPL>,-
          38 A5      04D7                    UCB$L_DEVCHAR(R5)        ; skip terminal if DECnet device
             1B  12  04D9            BNEQ    3$                       ; or spooled
             01  E0  04DB            BBS     #DEV$V_DET,-
       16 3C A5      04DD                    UCB$L_DEVCHAR2(R5),3$    ; skip terminal if detached
          50 A6  E0  04E0            BBS     BRK$L_REQID(R6),-
    0F 00A8 C5       04E3                    UCB$Q_TL_BRKTHRU(R5),3$ ; Or specific class disabled
             04  E0  04E7            BBS     #TT2$V_BRDCSTMBX,-
       0D 48 A5      04E9                    UCB$L_DEVDEPND2(R5),5$ ; must try this term if BRDCSTMBX
 00020001 8F  D3     04EC            BITL    #<TT$M_NOBRDCST!TT$M_PASSALL>,-
          44 A5      04F2                    UCB$L_DEVDEPEND(R5)      ; test for NOBROADCAST or PASSALL
```

```
              03   13 04F4           BEQL    5$                        ; try terminal if neither set
                      04F6           ;
                      04F6           ; For some reason, this device is not acceptable
                      04F6           ;
            004F   31 04F6      3$:  BRW     40$                       ; skip to next terminal
                      04F9
              02   E1 04F9      5$:  BBC     #BRK$V_CHKPRIV,-
        2E 66 A6      04FB                   BRK$B_STS(R6),30$         ; Branch if priv check not required
                      04FE           ;
                      04FE           ; Search up process tree to see if owner
                      04FE           ;
     51    1C A6   D0 04FE           MOVL    BRK$L_PCB(R6),R1          ; PCB address
     52    2C A5   D0 0502           MOVL    UCB$L_PID(R5),R2          ; owner PID
     52    60 A1   D1 0506     10$:  CMPL    PCB$L_PID(R1),R2          ; compare PIDs
              20   13 050A           BEQL    30$                       ; branch if OK
     51    1C A1   3C 050C           MOVZWL  PCB$L_OWNER(R1),R1        ; Get index of owner
              0A   13 0510           BEQL    20$                       ; If equal then none, must have priv
51   00000000'FF41 D0 0512           MOVL    @L^SCH$GL_PCBVEC[R1],R1   ; Get Owner PCB address
              EA   11 051A           BRB     10$                       ; Loop
                      051C     20$:
     54    1C A6   D0 051C           MOVL    BRK$L_PCB(R6),R4          ; PCB address
                      0520           IFPRIV  OPER,30$                  ; If privilege, ok to send message
  50     2894 8F   3C 0526           MOVZWL  #SS$_NOOPER,R0            ; set error
                   05 052B           RSB                               ; exit
                      052C           ;
                      052C           ; Set up name and unit number
                      052C           ;
                      052C     30$:
              57   DD 052C           PUSHL   R7                        ; Save R7
        50    0F   9A 052E           MOVZBL  #BRK$S_DEVNAME-1,R0       ; Size of buffer
     57    0C A6   9E 0531           MOVAB   BRK$T_DEVNAME(R6),R7      ; Address of buffer
     51    01 A7   9E 0535           MOVAB   1(R7),R1                  ; Address past byte count
        54    01   CE 0539           MNEGL   #1,R4                     ; Standard device name
     00000000'GF   16 053C           JSB     G^IOC$CVT_DEVNAM          ; convert to regular device name
           57 8ED0    0542           POPL    R7                        ; Restore R7
        09    50   E8 0545           BLBS    R0,50$                    ; skip this device if error
                      0548           ;
                      0548           ; This terminal failed, reset and loop
                      0548           ;
                      0548     40$:
            FC99   30 0548           BSBW    UNLOCK_DB                 ; unlock database
           76 A6   B6 054B           INCW    BRK$W_REFUSEDCNT(R6)      ; Increment
            FEA5   31 054E           BRW     GET_NEXT_TERMINAL         ; Loop
                      0551     50$:
     0C A6    51   90 0551           MOVB    R1,BRK$T_DEVNAME(R6)      ; Length of string
     58 A6    55   D0 0555           MOVL    R5,BRK$L_UCBCTX(R6)       ; save UCB address
                      0559           ;
                      0559           ; set up TRMNAME for mailbox message
                      0559           ;
           54 A5   B0 0559           MOVW    UCB$W_UNIT(R5),-
           7A A6      055C                   BRK$W_TRMUNIT(R6)         ; unit number
     50    28 A5   D0 055E           MOVL    UCB$L_DDB(R5),R0          ; Fetch DDB
           14 A0   7D 0562           MOVQ    DDB$T_NAME(R0),-
           7C A6      0565                   BRK$T_TRMNAME(R6)         ; set TRMNAME (first half)
           1C A0   7D 0567           MOVQ    DDB$T_NAME+8(R0),-
         0084 C6      056A                   BRK$T_TRMNAME+8(R6)       ; set TRMNAME (second half)
        50    01   9A 056D           MOVZBL  #SS$_NORMAL,R0            ; set success
```

```
                05 0570            RSB
                   0571
                   0571 LOCKDB:
            00  E2 0571            BBSS    #BRK$V_LOCKED,-
      0A 66 A6     0573                    BRK$B_STS(R6),10$           ; set locked flag
   54   1C A6   D0 0576            MOVL    BRK$L_PCB(R6),R4            ; Set PCB address
   00000000'GF  16 057A            JSB     G^SCH$IOLOCKR               ; lock I/O database for read access
                05 0580 10$:       RSB
                   0581
```

```
                              0581          .SBTTL  FIND_NEXT_TERM - Search I/O database
                              0581   ;++
                              0581   ;
                              0581   ; FUNCTIONAL DESCRIPTION:
                              0581   ;
                              0581   ;       Given the UCB context of the last terminal, find the next
                              0581   ;       terminal that qualifies. Terminal must be online.
                              0581   ;
                              0581   ;       If looking for all terminals, an unowned terminal is skipped
                              0581   ;       if autobauding.
                              0581   ;
                              0581   ; CALLING SEQUENCE:
                              0581   ;
                              0581   ;       BSBW    FIND_NEXT_TERM
                              0581   ;
                              0581   ; INPUT PARAMETERS:
                              0581   ;
                              0581   ;       R6 - BRK
                              0581   ;
                              0581   ; IMPLICIT INPUTS:
                              0581   ;       NONE
                              0581   ;
                              0581   ; OUTPUT PARAMETERS:
                              0581   ;
                              0581   ;       R5 - points to UCB
                              0581   ;
                              0581   ; COMPLETION CODES:
                              0581   ;
                              0581   ;       R0 = 1, R5 is UCB
                              0581   ;       R0 = 0, no more terminals
                              0581   ;
                              0581   ;       All other registers preserved.
                              0581   ;
                              0581   ; SIDE EFFECTS:
                              0581   ;       NONE
                              0581   ;
                              0581   ;--
                              0581
                              0581  FIND_NEXT_TERM:
                              0581
          0C00 8F     BB      0581          PUSHR   #^M<R10,R11>            ; Save
      5A    58 A6     7D      0585          MOVQ    BRK$L_UCBCTX(R6),R10    ; ucb and ddb pair
                              0589
               0C     13      0589          BEQL    20$                     ; *** TEMP
               50     D4      058B          CLRL    R0                      ; *** TEMP
30 AA   FFFFFFFF 8F   D1      058D          CMPL    #-1,UCB$L_LINK(R10)     ; *** TEMP until SCAN_IODB enhanced
               2F     13      0595          BEQL    40$                     ; *** TEMP  to handle missing UCBs
                              0597  20$:
       00000000'GF    16      0597          JSB     G^IOC$SCAN_IODB         ; Fetch next UCB
            26 50     E9      059D          BLBC    R0,40$                  ; branch if done
                              05A0   ;
                              05A0   ; Have valid UCB, see if it's a terminal
                              05A0   ;
               02     E1      05A0          BBC     #DEV$V_TRM,-
         F2 38 AA             05A2                  UCB$L_DEVCHAR(R10),20$  ; Get next if not terminal
               04     E1      05A5          BBC     #UCB$V_ONLINE,-
         ED 64 AA             05A7                  UCB$W_STS(R10),20$      ; next ucb if offline
```

```
        5C AA    B5  05AA             TSTW    UCB$W_REFC(R10)           ; terminal allocated?
           10    12  05AD             BNEQ    30$                       ; yes, do write
           04    B1  05AF             CMPW    #BRK$C_ALLTERMS,-
        4C A6        05B1                     BRK$W_SENDTYPE(R6)        ; for all terminals?
           E2    12  05B3             BNEQ    20$                       ; no, try next
           01    E1  05B5             BBC     #TT2$V_AUTOBAUD,-
     05 48 AA        05B7                     UCB$L_DEVDEPND2(R10),30$ ; branch if not autobaud
        76 A6    B6  05BA             INCW    BRK$W_REFUSEDCNT(R6)      ; Refused due to autobaud
           DB    11  05BD             BRB     20$                       ; try again
                     05BF
        55 5A    D0  05BF   30$:      MOVL    R10,R5                    ; Set output
     58 A6 5A    7D  05C2             MOVQ    R10,BRK$L_UCBCTX(R6)      ; save ucb and ddb pair
                     05C6
     0C00 8F     BA  05C6   40$:      POPR    #^M<R10,R11>              ; Restore
                 05  05CA             RSB                               ; Return (assumes R0 unmodified from
                     05CB                                               ;          call above)
                     05CB
                     05CB
```

```
                         05CB           .SBTTL  QIO_DONE - process qio completion
                         05CB
                         05CB    ;++
                         05CB    ;
                         05CB    ; FUNCTIONAL DESCRIPTION:
                         05CB    ;
                         05CB    ;       Completion AST routine for QIO to terminal.
                         05CB    ;
                         05CB    ; CALLING SEQUENCE:
                         05CB    ;
                         05CB    ;       CALLG (as an AST)
                         05CB    ;
                         05CB    ; INPUT PARAMETERS:
                         05CB    ;
                         05CB    ;       4(AP) - Address of per QIO context within BRK
                         05CB    ;
                         05CB    ; IMPLICIT INPUTS:
                         05CB    ;       NONE
                         05CB    ;
                         05CB    ; OUTPUT PARAMETERS:
                         05CB    ;       NONE
                         05CB    ;
                         05CB    ; IMPLICIT OUTPUTS:
                         05CB    ;       NONE
                         05CB    ;
                         05CB    ; COMPLETION CODES:
                         05CB    ;       NONE
                         05CB    ;
                         05CB    ; SIDE EFFECTS:
                         05CB    ;
                         05CB    ;       May result in another QIO being performed or
                         05CB    ;       completion of service.
                         05CB    ;
                         05CB    ;--
                         05CB
                   0FFC  05CB    QIO_DONE:      .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                         05CD
     57   04 AC    D0    05CD           MOVL    4(AP),R7                ; QIO context
       56    67    D0    05D1           MOVL    BRK2$L_COMMON(R7),R6    ; BRK common area
                         05D4
         2C A6     7D    05D4           MOVQ    BRK$Q_TIMEOUT(R6),-
         2C A6           05D7                   BRK$Q_TIMEOUT(R6)       ; Time out specified?
            0B     13    05D9           BEQL    20$                     ; branch if no
                         05DB           $CANTIM_S REQIDT = R7           ; Cancel timer
                         05E6    20$:
                         05E6           $DASSGN_S CHAN = BRK2$W_CHAN(R7) ; Deassign channel
                         05F1           ;
                         05F1           ; check IOSB
                         05F1           ;
     50   04 A7    3C    05F1           MOVZWL  BRK2$Q_IOSB(R7),R0      ; Fetch status
         11 50     E8    05F5           BLBS    R0,30$                  ; branch if no error
   50   0830 8F    B1    05F8           CMPW    #SS$_CANCEL,R0          ; Make sure it was cancel (from timeout)
            0D     13    05FD           BEQL    40$                     ;
     50     2C     B1    05FF           CMPW    #SS$_ABORT,R0           ; Make sure it was cancel (from timeout)
            08     13    0602           BEQL    40$                     ;
         76 A6     B6    0604           INCW    BRK$W_REFUSEDCNT(R6)    ; One more non-successful completion
            03     11    0607           BRB     40$                     ; continue
```

```
                0609     30$:
  72 A6    B6   0609                INCW    BRK$W_SUCCESSCNT(R6)      ; One more successful completion
                060C     40$:
  0A A6    B7   060C                DECW    BRK$W_OUTCNT(R6)          ; One less outstanding
   FBE5    30   060F                BSBW    DO_WRITE                  ; Do next write with this context
  02 50    E8   0612                BLBS    R0,100$                   ; branch if success
                0615
     01    10   0615                BSBB    CHECK_COMPLETE            ; check for completion
           04   0617     100$:      RET                               ; exit ast
```

```
                    0618         .SBTTL  CHECK_COMPLETE - Check completion criterion
                    0618   ;++
                    0618   ;
                    0618   ; FUNCTIONAL DESCRIPTION:
                    0618   ;
                    0618   ;       See if service is done with all it's duties and
                    0618   ;       complete if so.
                    0618   ;
                    0618   ; CALLING SEQUENCE:
                    0618   ;
                    0618   ;       BSBW    CHECK_COMPLETE
                    0618   ;
                    0618   ; INPUT PARAMETERS:
                    0618   ;
                    0618   ;       R6 - BRK
                    0618   ;
                    0618   ; IMPLICIT INPUTS:
                    0618   ;       NONE
                    0618   ;
                    0618   ; OUTPUT PARAMETERS:
                    0618   ;       NONE
                    0618   ;
                    0618   ; IMPLICIT OUTPUTS:
                    0618   ;       NONE
                    0618   ;
                    0618   ; COMPLETION CODES:
                    0618   ;       NONE
                    0618   ;
                    0618   ; SIDE EFFECTS:
                    0618   ;
                    0618   ;       R0, R1 destroyed
                    0618   ;
                    0618   ;--
                    0618
                    0618   CHECK_COMPLETE:
       0A A6    B5  0618           TSTW    BRK$W_OUTCNT(R6)         ; I/O still outstanding?
          01    13  061B           BEQL    10$                      ; branch if done
                05  061D           RSB                              ; otherwise, exit
                    061E           ;
                    061E           ; Return status and complete service
                    061E           ;
                    061E   10$:
51     20 A6    D0  061E           MOVL    BRK$L_IOSB(R6),R1        ; return IOSB
          13    13  0622           BEQL    30$                      ; Branch if none
    08 70 A6    E9  0624           BLBC    BRK$W_STATUS(R6),20$     ; Branch if other error occurred
       72 A6    B5  0628           TSTW    BRK$W_SUCCESSCNT(R6)     ; any messages sent?
          06    12  062B           BNEQ    20$                      ; branch if yes
  0084 8F       B0  062D           MOVW    #SS$_DEVOFFLINE,-
       70 A6        0631                   BRK$W_STATUS(R6)         ; set device off line
61     70 A6    7D  0633   20$:    MOVQ    BRK$W_STATUS(R6),(R1)    ; Return status and counts
                    0637           ;
                    0637           ; Deliver AST if necessary
                    0637           ;
                    0637   30$:
51     24 A6    D0  0637           MOVL    BRK$L_ASTADR(R6),R1      ; Fetch address
          12    13  063B           BEQL    40$                      ; Branch if no AST
50     67 A6    9A  063D           MOVZBL  BRK$B_PRVMODE(R6),R0     ; Set previous mode
```

```
                        0641      ; DESIGN NOTE: *** Should AST quota be taken at initiation of service?
                        0641      ;       If so - must use SCH$QAST here (to return quota).
                        0641      ;       Does this imply non-paged pool for ACB? Could be a problem.
                        0641      ;
                        0641              $DCLAST_S -
                        0641                      ASTADR = (R1),-                   ; AST routine
                        0641                      ASTPRM = BRK$L_ASTPRM(R6),-       ; AST parameter
                        0641                      ACMODE = R0                       ; access mode of caller
                        064F              ;
                        064F              ; Set Event Flag Number
                        064F              ;
                        064F      40$:
        51   64 A6   3C 064F              MOVZWL  BRK$W_EFN(R6),R1          ; Fetch number
                        0653              $SETEF_S EFN = R1                 ; Set efn
                        065C              ;
                        065C              ; Return storage
                        065C              ;
                        065C
                        065C      ; R6 - BRK
                        065C
                        065C      RETURN_MEMORY:
                        065C
                   50 DD 065C             PUSHL   R0                        ; Save
              50   56 D0 065E             MOVL    R6,R0                     ; Address of block
        51   08 A6   3C 0661              MOVZWL  BRK$W_SIZE(R6),R1         ; Size
        00000000'GF  16 0665              JSB     G^EXE$DEAP1               ; Deallocate
              50 8ED0    066B             POPL    R0                        ; Restore
                     05 066E              RSB                               ; Return
```

```
SYS
Sym

CAN
CCB
CCB
CCB
CCB
CHA
COD
DC$
DDT
DEV
DEV
DYN
EXE
EXE
EXE
EXE
EXE
IO$
IOC
IOC
IPL
IPL
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
NAR
PCB
PCB
PCB
PCB
PCB
PCB
PMS
PR$
PR$
RSN
RSN
SCH
SS$
SS$
SS$
SS$
UCB
UCB
UCB
UCB
UCB
```

```
                     066F            .SBTTL  QIO_TIMEOUT - process qio timeout
                     066F
                     066F   ;++
                     066F   ;
                     066F   ; FUNCTIONAL DESCRIPTION:
                     066F   ;
                     066F   ;
                     066F   ; CALLING SEQUENCE:
                     066F   ;        NONE
                     066F   ;
                     066F   ; INPUT PARAMETERS:
                     066F   ;
                     066F   ;        4(AP) - QIO context address
                     066F   ;
                     066F   ; IMPLICIT INPUTS:
                     066F   ;        NONE
                     066F   ;
                     066F   ; OUTPUT PARAMETERS:
                     066F   ;        NONE
                     066F   ;
                     066F   ; IMPLICIT OUTPUTS:
                     066F   ;        NONE
                     066F   ;
                     066F   ; COMPLETION CODES:
                     066F   ;        NONE
                     066F   ;
                     066F   ; SIDE EFFECTS:
                     066F   ;        NONE
                     066F   ;
                     066F   ;--
                     066F
               0040  066F   QIO_TIMEOUT:    .WORD   ^M<R6>
                     0671
  50   04 AC    D0   0671            MOVL    4(AP),R0                ; Fetch context
     56   60    D0   0675            MOVL    BRK2$L_COMMON(R0),R6    ; fetch common area address
     74 A6      B6   0678            INCW    BRK$W_TIMEOUTCNT(R6)    ; increment time out count ???
                     067B            $CANCEL_S BRK2$W_CHAN(R0)       ; Cancel I/O, wait for qio_done ast
                 04  0686            RET
                     0687
```

```
                            0687
                   007C     0687           .ENTRY  EXE$BRDCST, ^M<R2,R3,R4,R5,R6>  ; OLD SYS$BRDCST...
                            0689
  6D   00000000'GF    9E    0689           MOVAB   G^EXE$SIGTORET,(FP)        ; Set condition handler
                            0690
       51   04 AC     D0    0690           MOVL    4(AP),R1                   ; Get message address
                            0694           ;
                            0694           ; Figure out send type
                            0694           ;
          53   04     9A    0694           MOVZBL  #BRK$C_ALLTERMS,R3         ; Assume all terminals
       52   08 AC     D0    0697           MOVL    8(AP),R2                   ; Fetch descriptor address
               0A     13    069B           BEQL    20$                        ; Branch if all terminals
          53   03     9A    069D           MOVZBL  #BRK$C_ALLUSERS,R3         ; Assume all users
               62     D5    06A0           TSTL    (R2)                       ; Check length
               03     13    06A2           BEQL    20$                        ; Branch if zero
          53   01     9A    06A4           MOVZBL  #BRK$C_DEVICE,R3           ; Must be terminal name
                            06A7      20$:
               54     D4    06A7           CLRL    R4                         ; Clear R4 - no flags
          55   20     9A    06A9           MOVZBL  #^A/ /,R5                  ; Default carcon if only 2 parameters
          6C   04     D1    06AC           CMPL    #4,(AP)                    ; More parameters?
               04     12    06AF           BNEQ    30$                        ; Branch if no
       54   0C AC     7D    06B1           MOVQ    12(AP),R4                  ; Flags and carcon
                            06B5      30$:
          56   7E     7E    06B5           MOVAQ   -(SP),R6                   ; allocate IOSB on stack
                            06B8           $BRKTHRUW_S     -                  ; Call breakthru and wait
                            06B8                   EFN    = #BRK_C_BRDCSTEFN,-
                            06B8                   MSGBUF = (R1),-
                            06B8                   SENDTO = (R2),-
                            06B8                   SNDTYP = R3,-
                            06B8                   FLAGS  = R4,-
                            06B8                   CARCON = R5,-
                            06B8                   TIMOUT = #10,-             ; *** SYSGEN PARAMETER ???
                            06B8                   IOSB   = (R6)
            03 50     E9    06D3           BLBC    R0,60$                     ; Branch if error
          50   66     3C    06D6           MOVZWL  (R6),R0                    ; Use IOSB status
                            06D9      60$:
  50   00002894 8F    D1    06D9           CMPL    #SS$_NOOPER,R0             ; new status?
               03     12    06E0           BNEQU   70$                        ; nope, exit
          50   24     3C    06E2           MOVZWL  #SS$_NOPRIV,R0             ; set status
               04     06E5            70$: RET                                ; EXIT
                            06E6
                            06E6      .END
```

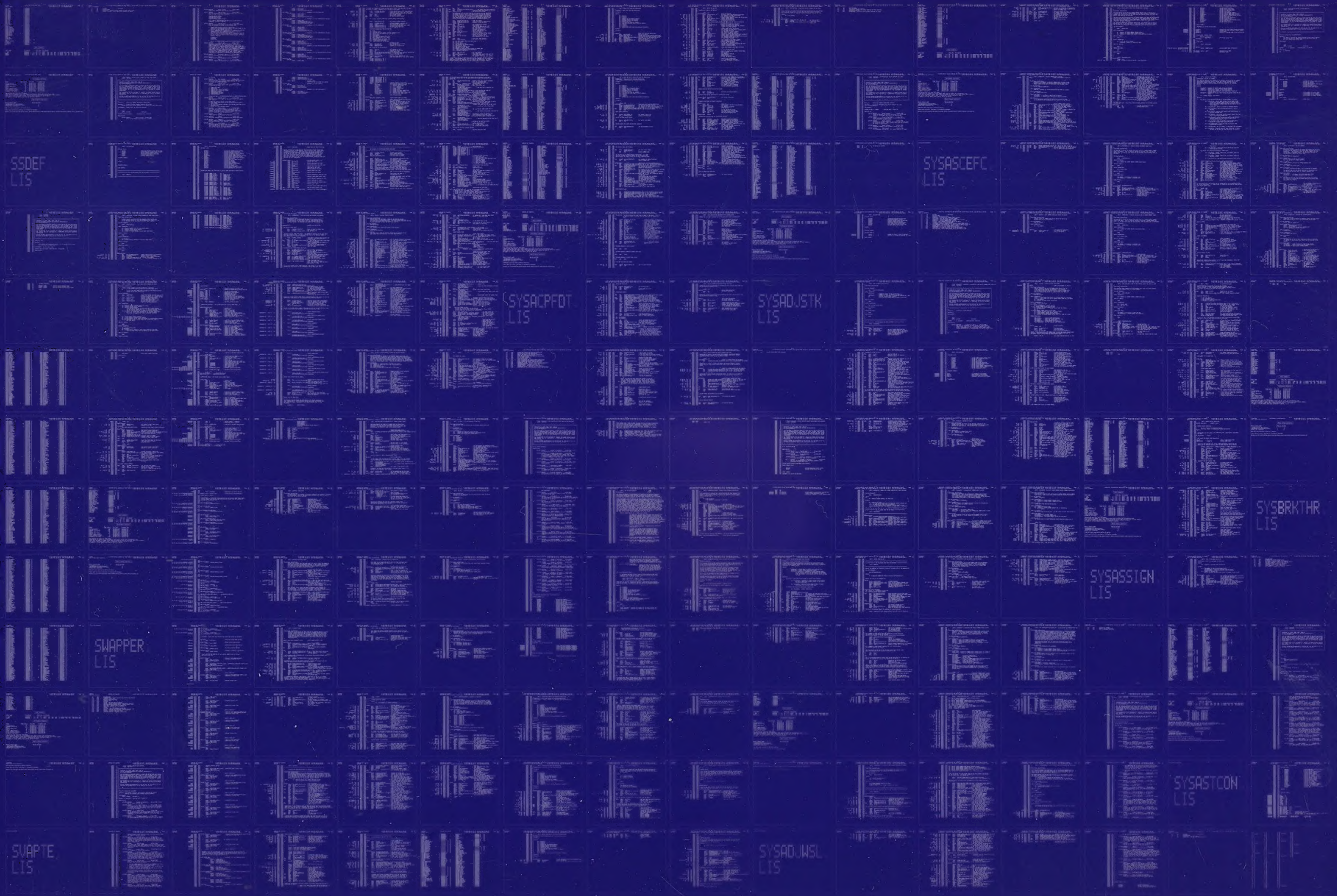
```
$$T1                            = 00000000                BRK$W_TRMUNIT                   = 0000007A
$$T2                            = 00000008                BRK2$C_LENGTH                   = 0000000E
ACCVIO_EXIT                       000000E4 R       02     BRK2$L_COMMON                   = 00000000
ALL_OK                            000000EF R       02     BRK2$Q_IOSB                     = 00000004
ALL_TERMS                         00000490 R       02     BRK2$W_CHAN                     = 0000000C
ASTADR                          = 00000028                BRK_C_BRDCSTEFN                 = 0000001F
ASTPRM                          = 0000002C                BRK_C_CLUTIMEOUT                = 0000000F
BADPARAM_EXIT                     000000DF R       02     BRK_C_DVIEFN                    = 0000001F
BRK$B_PRVMODE                   = 00000067                BRK_C_JPIEFN                    = 0000001F
BRK$B_STS                       = 00000066                BRK_C_MAXLINES                  = 00000018
BRK$C_ALLTERMS                  = 00000004                BRK_C_MINTIME                   = 00000004
BRK$C_ALLUSERS                  = 00000003                BRK_C_QIOEFN                    = 0000001F
BRK$C_DEVICE                    = 00000001                BRK_C_SIMULCAST                 = 00000004
BRK$C_LENGTH                    = 0000008E                BRK_C_TIMEFN                    = 0000001F
BRK$C_MAXSENDTYPE               = 00000004                CARCON                          = 00000018
BRK$L_ASTADR                    = 00000024                CCB$B_STS                       = 00000008
BRK$L_ASTPRM                    = 00000028                CCB$M_IMGTMP                    = 00000002
BRK$L_CARCON                    = 00000034                CHECK_COMPLETE                    00000618 R       02
BRK$L_DDBCTX                    = 0000005C                CLU$GL_CLUB                       ********   X     02
BRK$L_FLAGS                     = 00000038                CTL$GL_CCBBASE                    ********   X     02
BRK$L_IOSB                      = 00000020                DDB$S_NAME                      = 00000010
BRK$L_PCB                       = 0000001C                DDB$T_NAME                      = 00000014
BRK$L_PIDCTX                    = 00000054                DEV$M_NET                       = 00002000
BRK$L_QIOCTX                    = 00000060                DEV$M_SPL                       = 00000040
BRK$L_REQID                     = 00000050                DEV$V_AVL                       = 00000012
BRK$L_SCRMSG                    = 0000006C                DEV$V_DET                       = 00000001
BRK$L_SCRMSGLEN                 = 00000068                DEV$V_TRM                       = 00000002
BRK$L_UCBCTX                    = 00000058                DO_WRITE                          000001F7 R       02
BRK$M_CHKPRIV                   = 00000004                DVI$_DEVNAM                     = 00000020
BRK$M_DONE                      = 00000002                EFN                             = 00000004
BRK$Q_PRIVS                     = 00000000                ERASE_PAT                         00000000 R       02
BRK$Q_TIMEOUT                   = 0000002C                ERROR_EXIT                        000000E7 R       02
BRK$S_DEVNAME                   = 00000010                EXE$ALOP1IMAG                     ********   X     02
BRK$S_SENDNAME                  = 00000010                EXE$BRDCST                        00000687 RG      02
BRK$S_TRMNAME                   = 00000010                EXE$BRKTHRU                       00000025 RG      02
BRK$T_DEVNAME                   = 0000000C                EXE$CSP_BRKTHRU                   ********   X     02
BRK$T_MSGBUF                    = 0000008E                EXE$DEAP1                         ********   X     02
BRK$T_SENDNAME                  = 0000003C                EXE$PROBER_DSC                    ********   X     02
BRK$T_TRMNAME                   = 0000007C                EXE$SIGTORET                      ********   X     02
BRK$V_BOTTOM                    = 00000009                EXE$WRTMAILBOX                    ********   X     02
BRK$V_CHKPRIV                   = 00000002                FIND_NEXT_TERM                    00000581 R       02
BRK$V_CLUSTER                   = 0000000B                FLAGS                           = 0000001C
BRK$V_DONE                      = 00000001                GET_NEXT_TERMINAL                 000003F6 R       02
BRK$V_LOCKED                    = 00000000                GET_SENDTO                        00000339 R       02
BRK$V_NOREFRESH                 = 0000000A                HAVE_NAME                         000004A9 R       02
BRK$V_SCREEN                    = 00000008                HAVE_UCB                          000004C9 R       02
BRK$W_EFN                       = 00000064                IO$M_BREAKTHRU                  = 00000200
BRK$W_MSGLEN                    = 0000008C                IO$M_CANCTRLO                   = 00000040
BRK$W_OUTCNT                    = 0000000A                IO$M_REFRESH                    = 00002000
BRK$W_REFUSEDCNT                = 00000076                IO$_WRITEVBLK                   = 00000030
BRK$W_SECONDS                   = 0000004E                IOC$CVT_DEVNAM                    ********   X     02
BRK$W_SENDTYPE                  = 0000004C                IOC$SCAN_IODB                     ********   X     02
BRK$W_SIZE                      = 00000008                IOC$SEARCHDEV                     ********   X     02
BRK$W_STATUS                    = 00000070                IOSB                            = 00000014
BRK$W_SUCCESSCNT                = 00000072                JIB$S_USERNAME                  = 0000000C
BRK$W_TIMEOUTCNT                = 00000074                JIB$T_USERNAME                  = 0000000C
BRK$W_TRMMSG                    = 00000078                JPI$_TERMINAL                   = 0000031D
```

```
JPI$_USERNAME          = 00000202                  SYS$DASSGN             ******** GX  02
LOCKDB                   00000571 R     02         SYS$DCLAST             ******** GX  02
MSG$_TRMBRDCST           ********   X   02         SYS$FAO                ********  X  02
MSGBUF                 = 00000008                  SYS$GETDVIW            ******** GX  02
NEXT_TERM_ERROR          0000049B R     02         SYS$GETJPI             ******** GX  02
NO_MORE_TERM             000004A4 R     02         SYS$QIO                ******** GX  02
PCB$L_JIB              = 00000080                  SYS$SETAST             ******** GX  02
PCB$L_OWNER            = 0000001C                  SYS$SETEF              ******** GX  02
PCB$L_PHD              = 0000006C                  SYS$SETIMR             ******** GX  02
PCB$L_PID              = 00000060                  SYS$SETPRV             ******** GX  02
PCB$Q_PRIV             = 00000084                  TERM_DONE              0000049F R   02
PHD$Q_PRIVMSK          = 00000000                  TIMOUT               = 00000024
PR$_IPL                  ********   X   02         TT$M_NOBRDCST        = 00020000
PRV$M_BYPASS           = 20000000                  TT$M_PASSALL         = 00000001
PRV$M_SHARE            = 80000000                  TT2$V_AUTOBAUD       = 00000001
PRV$V_BYPASS           = 0000001D                  TT2$V_BRDCSTMBX      = 00000004
PRV$V_OPER             = 00000012                  TT2$V_DECCRT         = 0000001D
PRV$V_SHARE            = 0000001F                  UCB$L_AMB            = 00000060
PSL$S_PRVMOD           = 00000002                  UCB$L_DDB            = 00000028
PSL$V_PRVMOD           = 00000016                  UCB$L_DEVCHAR        = 00000038
QIO_DONE                 000005CB R     02         UCB$L_DEVCHAR2       = 0000003C
QIO_TIMEOUT              0000066F R     02         UCB$L_DEVDEPEND      = 00000044
REQID                  = 00000020                  UCB$L_DEVDEPND2      = 00000048
RETURN_MEMORY            0000065C R     02         UCB$L_LINK           = 00000030
SCH$CLREF                ********   X   02         UCB$L_PID            = 0000002C
SCH$GL_PCBVEC            ********   X   02         UCB$Q_TL_BRKTHRU     = 000000A8
SCH$IOLOCKR              ********   X   02         UCB$V_ONLINE         = 00000004
SCH$IOUNLOCK             ********   X   02         UCB$W_REFC           = 0000005C
SCREEN_CTRSTR            00000008 R     02         UCB$W_STS            = 00000064
SENDTO                 = 0000000C                  UCB$W_UNIT           = 00000054
SENDTYPE               = 00000010                  UNLOCK_DB              000001E4 R   02
SS$_ABORT              = 0000002C
SS$_ACCVIO             = 0000000C
SS$_BADPARAM           = 00000014
SS$_CANCEL             = 00000830
SS$_DEVOFFLINE         = 00000084
SS$_NOMOREPROC         = 000009A8
SS$_NOOPER             = 00002894
SS$_NOPRIV             = 00000024
SS$_NORMAL             = 00000001
STK$C_LEN                0000002C
STK$L_ENDLIST            00000018
STK$L_TERMLENR           00000014
STK$L_TERMNAME           00000010
STK$L_USERLENR           00000008
STK$L_USERNAME           00000004
STK$T_USERNAME           0000001E
STK$W_TERMJPI            0000000E
STK$W_TERMLEN            0000002A
STK$W_TERMSIZ            0000000C
STK$W_USERJPI            00000002
STK$W_USERLEN            0000001C
STK$W_USERSIZ            00000000
SYS$ASSIGN               ******** GX    02
SYS$BRKTHRUW             ******** GX    02
SYS$CANCEL               ******** GX    02
SYS$CANTIM               ******** GX    02
```

```
                                                        +-------------------+
                                                        ! Psect synopsis !
                                                        +-------------------+

PSECT name                         Allocation          PSECT No.  Attributes
----------                         ----------          ---------  ----------
.  ABS  .                          00000000  (     0.)  00 (   0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                              0000002C  (    44.)  01 (   1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
Y$EXEPAGED                         000006E6  (  1766.)  02 (   2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                                     +----------------------------+
                                                     ! Performance indicators !
                                                     +----------------------------+

Phase                         Page faults    CPU Time       Elapsed Time
-----                         -----------    --------       ------------
Initialization                         29    00:00:00.07    00:00:01.77
Command processing                    112    00:00:00.50    00:00:04.76
Pass 1                                623    00:00:27.10    00:01:22.88
Symbol table sort                       0    00:00:04.50    00:00:12.69
Pass 2                                220    00:00:05.39    00:00:20.48
Symbol table output                    24    00:00:00.21    00:00:00.42
Psect synopsis output                   2    00:00:00.03    00:00:00.22
Cross-reference output                  0    00:00:00.00    00:00:00.00
Assembler run totals                 1012    00:00:37.82    00:02:03.24

The working set limit was 2100 pages.
155190 bytes (304 pages) of virtual memory were used to buffer the intermediate code.
There were 150 pages of symbol table space allocated to hold 2771 non-local and 66 local symbols.
1317 source lines were read in Pass 1, producing 24 object records in Pass 2.
53 pages of virtual memory were used to define 51 macros.

                                                    +------------------------------+
                                                    ! Macro library statistics !
                                                    +------------------------------+

Macro library name                                  Macros defined
------------------                                  --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                                  15
_$255$DUA28:[SYSLIB]STARLET.MLB;2                               32
TOTALS (all libraries)                                          47

3023 GETS were required to define 47 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYSBRKTHR/OBJ=OBJ$:SYSBRKTHR MSRC$:SYSBRKTHR/UPDATE=(ENH$:SYSBRKTHR)+EXECML$/LIB
```

0381 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



SSDEF
LIS

SYSASCEFC
LIS

SYSACPFDT
LIS

SYSADJSTK
LIS

SYSBRKTHR
LIS

SYSASSIGN
LIS

SWAPPER
LIS

SYSASTCON
LIS

SVAPTE
LIS

SYSADJWSL
LIS

0382 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY